Markdown 简单的世界

wizardforcel

Published with GitBook



目錄

Markdown - 简单的世界

Markdown 简介

Markdown是一种可以使用普通文本编辑器编写的标记语言，通过简单的标记语法，它可以使普通文本内容具有一定的格式。

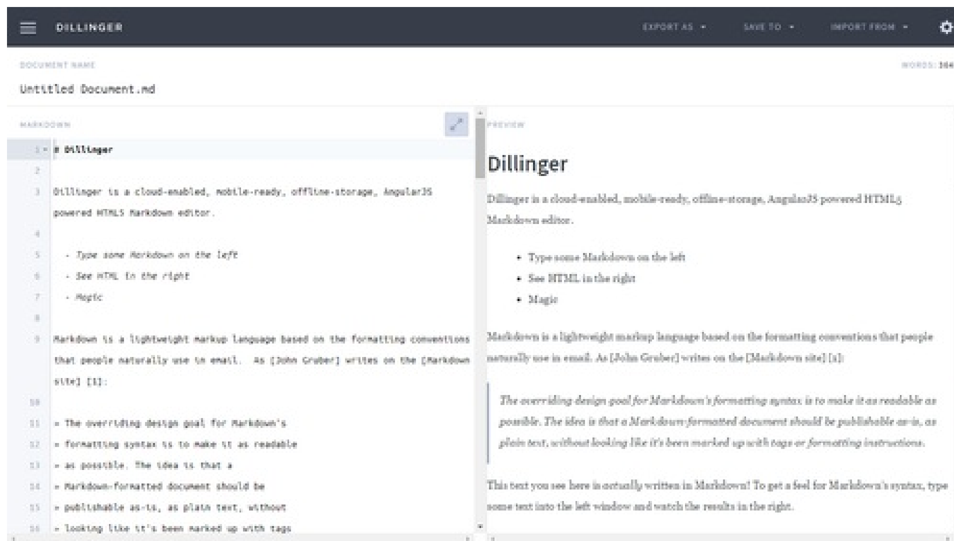
Markdown具有一系列衍生版本，用于扩展Markdown的功能（如表格、脚注、内嵌HTML等等），这些功能原初的Markdown尚不具备，它们能让Markdown转换成更多的格式，例如LaTeX，Docbook。Markdown增强版中比较有名的有Markdown Extra、MultiMarkdown、Maruku等。这些衍生版本要么基于工具，如Pandoc；要么基于网站，如GitHub和Wikipedia，在语法上基本兼容，但在一些语法和渲染效果上有改动。

Markdown 编辑器推荐

在线版

1. dillinger

漂亮强大，支持md, html, pdf 文件导出。支持dropbox, onedrive，google drive, github. 来自国外，可能不够稳定。



2. StackEdit

输出美观大方，可本地保存，还有拼写检查，但是对中文支持不好，可以从截图中看到，中文全部被标记为拼写错误，而且源代码中汉字的字间距太大。



3. MaHua

小众软件推荐，界面有些简陋。



4. 简书

一个很不错的博客平台，每几秒钟便会自动存入一个备份。可以直接从本地拖入照片生成链接，一直在不断优化。作为一个博客平台，需要注册账号后方能进行写作。



5. 马克飞象

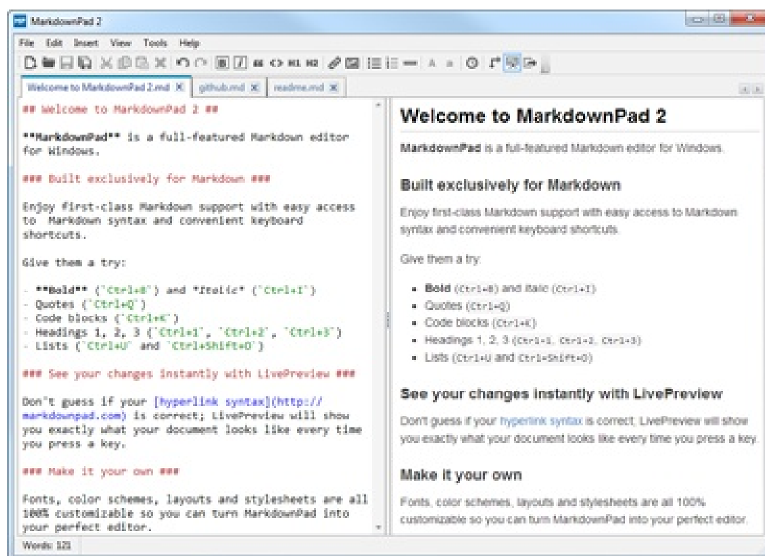
因为印象笔记不支持Markdown，而这款可以直接把文本存到印象笔记的编辑器对于重度印象笔记用户是个不错的选择。付费软件，可以免费试用。



windows

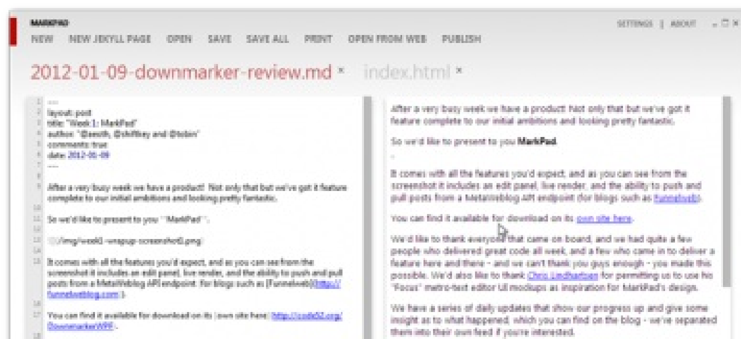
1. MarkdownPad

用户可以通过键盘快捷键和工具栏按钮来使用或者移除 Markdown 格式。MarkdownPad左右栏的分割方式令用户可以实时看到 HTML 格式的 Markdown 文档。



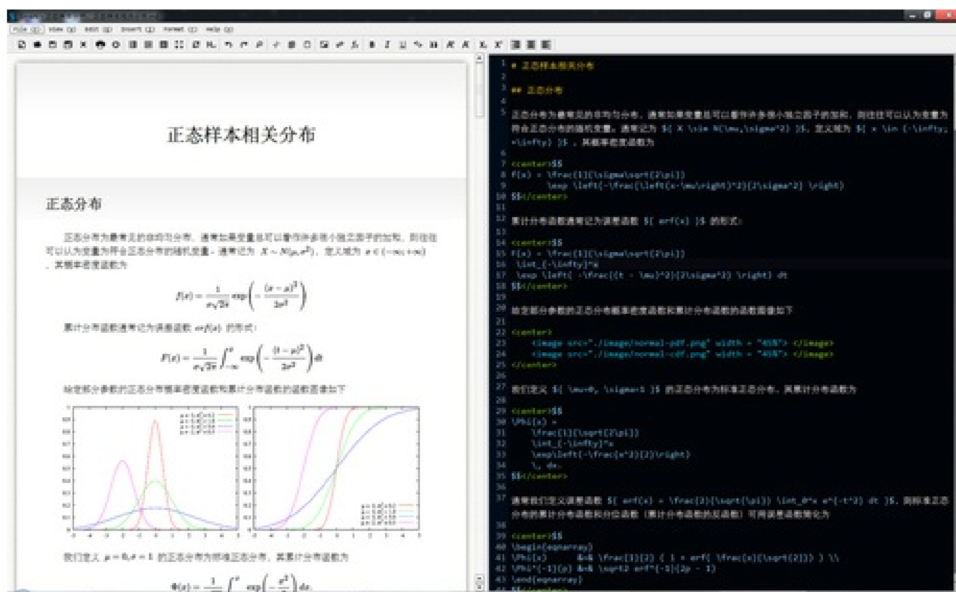
2. MarkPad

MarkPad 是款开源的 Markdown 编辑器，与 Window 8 风格和谐友好的界面，可以直接在你的博客或者 GitHub 中打开、保存文档，直接将图片粘贴到 Markdown 文档中。



3. Smark

开源软件。



4. Miu

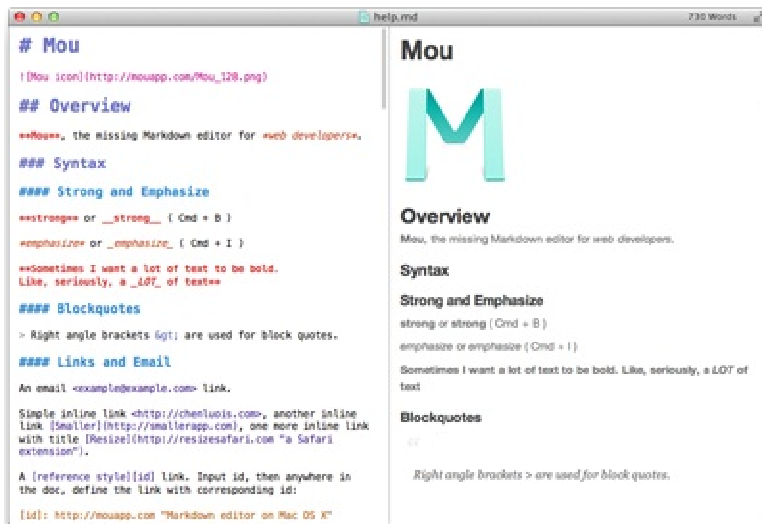
一款模仿mou 的windows平台markdown编辑器，小众推荐，必属精品，官网无法打开，好在小众提供了百度云下载。界面美观，功能不够成熟



OSX

1. Mou

Mou 是 Mac下杰出的 Markdown 编辑器，提供语法高亮、在线预览、同步滚动、全屏模式，支持自定保存、自动匹配，允许自定义主题，支持 CSS，HTML 和 PDF 导出等。



2. MacDown

开源且免费。



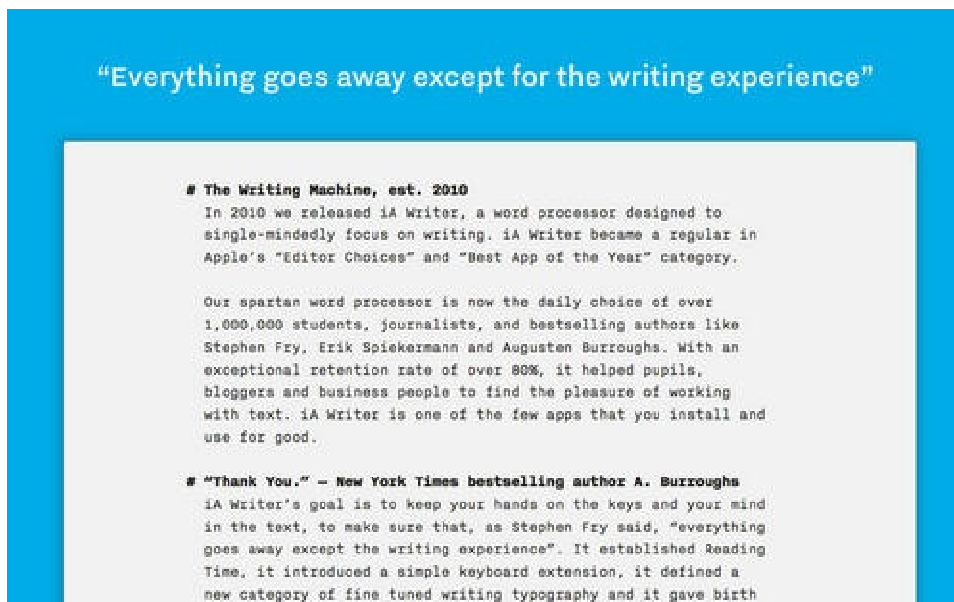
3. Ulysses

支持OS X , iPad， 售价 283元。堪称markdown编辑器中的佼佼者。



4. iA Writer

支持OS X , iOS 及Android，未优化中文显示，售价68，pro版本128元。



5. MWeb

专业的 Markdown 编辑器。UI漂亮，主题可选。支持markdown扩展语法，支持打字机滚动模式，支持发布到wp、blogger、tumblr等多个博客。



跨平台

1. Cmd Markdown

作业部落出品，也是一款不错的工具和博客平台兼顾的产品。全平台且提供web版。



2. 小书匠编辑器

全平台覆盖并且有web版。



3. FarBox

一个支持Markdown写作语法的博客平台，让用户通过Dropbox（现在默认是自己的同步服务器）直接建立个人网站。FarBox编辑器免费，支持多平台（无web版，Linux版停止维护），个人认为是Windows平台最优雅的编辑器。多说一句，Farbox服务可以免费试用，在本地编辑器内写作自动同步发布在个人博客，对像笔者这样希望有个人博客但却不愿折腾的小白来说，是个不错的选择。



4. Sublime Text 2

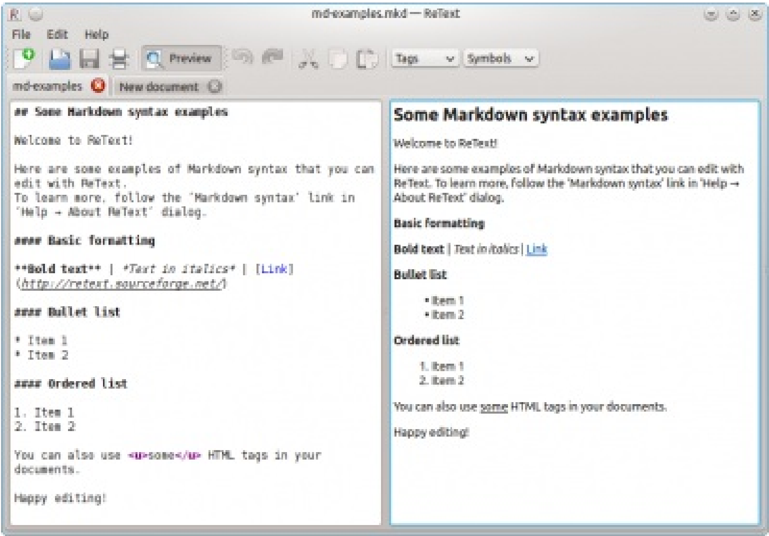
界面简约大方，定位专业。价格70美元，但是如果你能忍受时不时弹出的注册提醒，完全可以免费使用。唯一美中不足的是markdown并非其原生功能，需要安装插件。

5. Atom

github出的编辑器，功能十分强大，除了编写代码之外还可以做为markdown编辑器，并且原生支持markdown预览，按 `ctrl+shift+m` 弹出。

6. ReText

用python开发的，所以跨平台。ReText 是一个使用 Markdown 语法和 reStructuredText (reST) 结构的文本编辑器，编辑的内容支持导出到 PDF、ODT 和 HTML 以及纯文本，支持即时预览、网页生成以及 HTML 语法高亮、全屏模式，可导出文件到 Google Docs 等。



注

部分内容来自月光博客 - 好用的Markdown编辑器一览。

Markdown 基本语法

段落

非常自然，一行文字就是一个段落。

比如

```
这是一个段落。
```

会被解释成

```
<p>这是一个段落。</p>
```

如果你需要另起一段，请在两个段落之间隔一个空行。

```
这是一个段落。

这是另一个段落。
```

会解释成

```
<p>这是一个段落<p>
<p>这是另一个段落</p>
```

不隔一个空行的换行行为，在一些编辑器中被解释为换行，即插入一个 `<br />` 标签。对与另外一些编辑器，会被解释为插入一个空格。对于后者，若要插入换行标签，请在当前一行的结尾打两个空格。

粗体、斜体

可以使用星号 `*` 或下划线 `_` 指定粗体或者斜体。

```
*这是斜体*
_这也是斜体_
**这是粗体**
***这是粗体+斜体***
```

会被解释成

```
<em>这是斜体</em>
<em>这也是斜体</em>
<strong>这是粗体</strong>
<strong><em>这是粗体+斜体</strong></em>
```

删除线

一部分编辑器支持删除线，它不是经典markdown中的要素。用波浪线 `~` 定义删除线。

```
~~就像这样~~
```

会被解释成

```
<strike>就像这样</strike>
```

标题

markdown总支持1~6六级标题，通过在一行之前加上不同数量的井号来表示。

```
# 这是 H1 #

## 这是 H2 ##

### 这是 H3 ###

...

###### 这是 H6 ######
```

行尾可以加上任意数量的井号字符，这些字符不会算作标题内容。通常会加上相等数量的字符以保持对称。

此外，H1和H2也可以采用在文本下方添加底线来实现，比如：

```
这是 H1
=======

这是 H2
-------
```

引用

通过在行首加上大于号 `>` 来添加引用格式。

```
> This is a blockquote with two paragraphs. Lorem ipsum dolor sit amet,
consectetuer adipiscing elit. Aliquam hendrerit mi posuere lectus.
Vestibulum enim wisi, viverra nec, fringilla in, laoreet vitae, risus.

> Donec sit amet nisl. Aliquam semper ipsum sit amet velit. Suspendisse
id sem consectetuer libero luctus adipiscing.
```

引用可以嵌套：

```
> This is the first level of quoting.
>
> > This is nested blockquote.
>
> Back to the first level.
```

也可以嵌套其他格式：

```
> ## 这是一个标题。
>
> 1.   这是第一行列表项。
> 2.   这是第二行列表项。
>
> 给出一些例子代码：
>
>     return shell_exec("echo $input | $markdown_script");
```

列表

无序列表使用星号、加号或是减号作为列表标记：

```
*   Red
*   Green
*   Blue
```

等同于

```
+   Red
+   Green
+   Blue
```

和

```
-   Red
-   Green
-   Blue
```

有序列表则使用数字接着一个英文句点：

```
1.  Bird
2.  McHale
3.  Parish
```

数字并不会影响输出的 HTML 结果，也就是说上面的例子等同于：

```
1.  Bird
1.  McHale
1.  Parish
```

内联代码

用反引号 `` ` `` 来标记内联代码，它们会解释成 `<code>` 标签。如果代码的内容中有反引号，请用两个反引号包裹。代码中的 `&` 、 `<` 、 `>` 符号都会自动转义，请放心使用。

代码区域

有两种方式标记代码区域，原生风格是行首缩进死个空格。

```
这是一个普通段落：

    这是一个代码区块。
```

会被解释成

```
<p>这是一个普通段落：</p>

<pre><code>这是一个代码区块。
</code></pre>
```

除了行首的4个空格会被移出，其它不变。像内联代码一样，上述三种符号也会被转义。但在代码段中，星号之类的markdown标记符号则不会解析。

还有一种是github的风格，代码段的前后用三个反引号独占一行来标记。

````
```
这是一个代码区块。
```
````

目前主流编辑器都支持这种风格。

分隔线

你可以在一行中用三个以上的星号、减号、底线来建立一个分隔线，行内不能有其他东西。你也可以在星号或是减号中间插入空格。下面每种写法都可以建立分隔线：

```
* * *
***
*****
- - -
---------------------------------------
```

链接

```
[an example](http://example.com/)
[an example](http://example.com/ "Optional Title")
```

会被解释为

```
<a href='http://example.com/'>an example</a>
<a href='http://example.com/' title="Optional Title">an example</a>
```

除了上面的行内式，也可以使用参考式：

```
[an example][id]
```

然后在任意空白位置定义：

```
[id]: http://example.com/ "Optional Title"
```

图像

```
![Alt text](/path/to/img.jpg)
![Alt text](/path/to/img.jpg "Optional Title")
```

会被解释为

```
<img src='/path/to/img.jpg' alt='Alt text' />
<img src='/path/to/img.jpg' alt='Alt text' title='Optional Title' />
```

同样，图像也有类似的参考式语法。

自动链接

如果链接的地址和名字重复，可以用尖括号语法将其简化。

```
<http://example.com/>
```

就相当于

```
[http://example.com/](http://example.com/)
```

切记，大多数编辑器都会自动将符合url规则的东西视为链接，并且解释成链接。很多时候作者由于疏忽等缘故，链接和后面的中文之间缺少空格，导致链接不正常。所以我建议，链接要么加上尖括号，要么两端加上空格。

转义

markdown支持在以下字符前面插入反斜杠

```
\    反斜线
`    反引号
*    星号
_    底线
{}   花括号
[]   方括号
()   括弧
#    井字号
+    加号
-    减号
.    英文句点
!    惊叹号
```

插入之后，将不再解析这些字符，而是原样输出。

表格

表格是github风格独有的语法，但近年来渐渐被大多数编辑器支持。

```
| Item     | Value | Qty   |
| :------- | ----: | :---: |
| Computer | $1600 |  5    |
| Phone    | $12   |  12   |
| Pipe     | $1    |  234  |
```

会被解释成

```
<table>
<thead>
<tr>
  <th align="left">Item</th>
  <th align="right">Value</th>
  <th align="center">Qty</th>
</tr>
</thead>
<tbody><tr>
  <td align="left">Computer</td>
  <td align="right">$1600</td>
  <td align="center">5</td>
</tr>
<tr>
  <td align="left">Phone</td>
  <td align="right">$12</td>
  <td align="center">12</td>
</tr>
<tr>
  <td align="left">Pipe</td>
  <td align="right">$1</td>
  <td align="center">234</td>
</tr>
</tbody></table>
```

要注意第二行的冒号决定了居左居右还是居中，如果你不加冒号，默认是居左的。

另外可以把第一行去掉，做成没有表头的表格，但第二行始终是要有的。

内联 HTML

markdown 的语法简洁，但有其局限性，所以特意保留了内联html这种方式。任何html标签及其内容，都会原样输出到结果中。也就是说，标签中的星号等作为markdown结构的符号，以及构成html标签和实体的符号，都不会做任何转义。

Markdown 高级语法

只有少数编辑器支持，使用前请先确认。

定义列表

```
Term 1
Term 2
:   Definition A
:   Definition B
```

会被编译成

```
<dl>
<dd>Term 1</dd>
<dd>Term 2</dd>
<dt>Definition A<dt>
<dt>Definition A<dt>
</dl>
```

目录

通过 `[TOC]` 标记来插入目录。

TeX公式

内联的TeX公式使用一个美元符号标记。

```
$\Gamma(n) = (n-1)!\quad\forall n\in\mathbb N$
```

会被编译成

$\Gamma(n) = (n-1)!\quad\forall n\in\mathbb N$

TeX公式块用独占一行的两个美元符号来标记。

```
$$
\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt\,.
$$
```

会被编译成

$$\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt\,.$$

如果你的编辑器不支持这个功能，可以手动解决。首先引入mathjax脚本：

```
<script type="text/javascript" src="https://cdn.mathjax.org/mathjax/latest/MathJax.js?con
```

之后，在需要插入公式的地方使用 `<script>` 标签包裹公式：

```
<script type="math/tex">\Gamma(n) = (n-1)!\quad\forall n\in\mathbb N</script>

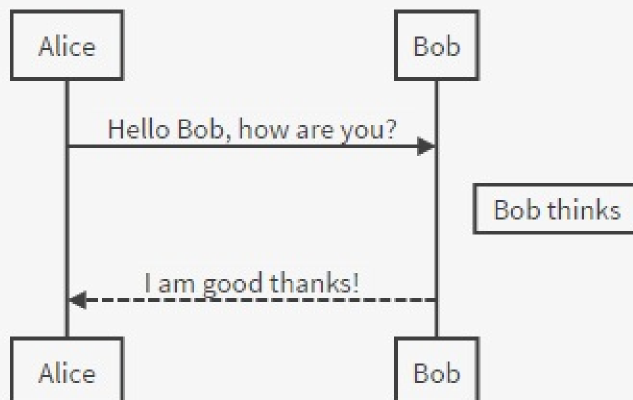
<script type="math/tex; mode=display">
\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt\,.
</script>
```

TeX的语法参考请见这里。

UML图

可以像这样来画uml时序图：

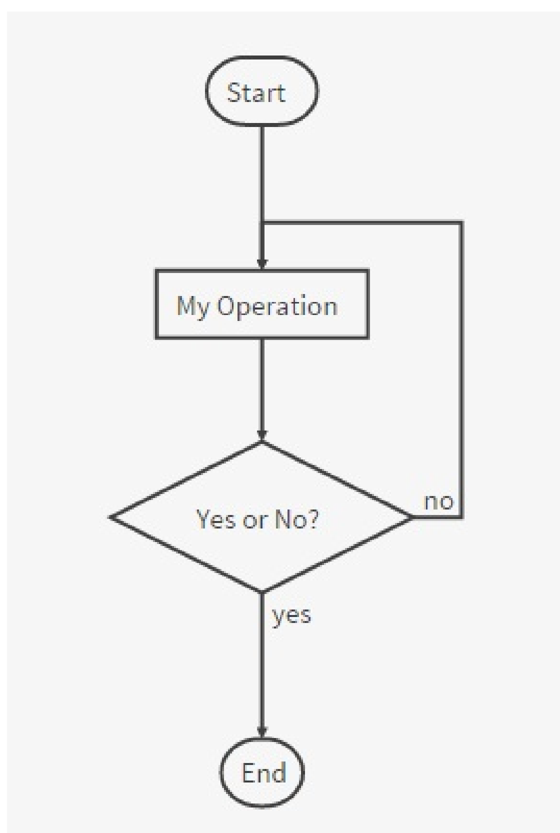
````
```sequence
Alice->Bob: Hello Bob, how are you?
Note right of Bob: Bob thinks
Bob-->Alice: I am good thanks!
```
````



这是uml流程图：

````
```flow
st=>start: Start
e=>end
op=>operation: My Operation
cond=>condition: Yes or No?

st->op->cond
cond(yes)->e
cond(no)->op
```
````



时序图的语法请见这里。流程图的语法请见这里。

Markdown + Gitbook

使用Gitbook制作电子书

Gitbook是一个命令行工具，可以把你的Markdown文件汇集成电子书，并提供PDF等多种格式输出。你可以把Gitbook生成的HTML发布出来，就形成了一个简单的静态网站。Gitbook还有一个同名的平台（gitbook.io），可以发布和销售电子书，并提供了一个Markdown客户端工具（支持Mac、Windows和Linux）帮助写作。以下是我在使用Gitbook中的笔记。

首先Gitbook和Git/Github都没有什么关系。它只是一个build book的工具而已。但它的Git前缀的确引起了许多人的迷惑，起初我认为至少它也是个和Github类似的Git平台吧，但其实没什么关系，你只要懂几条markdown语法，不必理解任何与Git相关的东西就能用Gitbook了，不要为其名字迷惑。

第**0**步 安装npm（Node Package Manager）。从node.js的官网上下载安装程序，即可完成Node.js和npm的安装。

第**1**步 通过npm安装Gitbook。

> $ npm install gitbook -g

完成后花10分钟阅读下Gitbook的帮助文档。如果你没耐心看手册，那就继续往下读吧 :D

第**2**步 了解Gitbook的基本规则。

Gitbook需要2个基本文件：

- README.md
- SUMMARY.md

README.md是关于你的书的介绍，而SUMMARY.md中则包含了书目，即章节结构，它的格式大致是：

```
*  [第1章](c1.md)
  *  [第1节](c1s1.md)
  *  [第2节](c1s2.md)
*  [第2章](c2.md)
```

剩下的东西就很好理解了，你只需要编写相应章节即可。在编辑完README.md和SUMMARY.md后，你可以运行以下命令：

> $ gitbook serve -p 8080 .

Gitbook首先把你的Markdown文件编译为HTML文件，并根据SUMMARY.md生成书的目录。所有生存的文件都保存在当前目录下的一个名为_book的子目录中。完成这些工作后，Gitbook会作为一个HTTP Server运行，并在8080端口监听HTTP请求。

运行以上命令后，打开浏览器，在地址栏输入：`http://localhost:8080` 即可看到你的书页了。

其中位于左侧书目顶部的 `Introduction` 一节就编译自README.md，而书目本身自编译自SUMMARY.md。你要在自己的网站上发布新书，只需把_book目录复制到服务器相应目录即可。至此Gitbook的基本用法就介绍完毕。下面简单讨论下Gitbook的其他应用，包括Gitbook的插件、与Github的融合、Gitbook客户端、Gitbook平台，以及Gitbook的问题。

Gitbook的插件支持

Gitbook可以生成HTML，因此它支持一些外部的JavaScript文件嵌入到HTML中，例如Google统计、Disqus评论系统等。以下以页面中嵌入Disqus评论为例。

首先是安装Gitbook的Disqus插件。

> $ npm install gitbook-plugin-disqus

然后建立一个book.json文件，其格式如下：

```
{  "plugins":  ["disqus"],  "pluginsConfig":  {  "disqus":  {  "shortName":  "NAME-FROM-D
```

把上面的 `NAME-FROM-DISQUS` 修改为你在Disqus上的项目名即可。

再次运行命令：

> $ gitbook serve -p 8080 .

并刷新浏览器，即可看到附加了Disqus评论的页面。

与**Github**的融合

Gitbook的博客上说Github提供了对Gitbook的特殊支持，但我没有测试。只是依然把源文件保存在Github上，然后用Gitbook去编译。期待Gitbook做的更好。

Gitbook客户端

Gitbook客户端支持Mac、Windows、Linux。我在Mac和Windows简单尝试了这个客户端，总体而言可以用。但也仅仅是可以用而已。你可以在客户端里编辑Markdown文件，并提供一个实时的预览窗口；可以关联到你的Gitbook账户，并把内容同步到gitbook.io，并为你生成PDF等。说句题外话，如果你要Markdown的客户端的话，飞象马克更好用，至少Vim编辑模式你得支持啊。

Gitbook的问题

Gitbook网站的访问速度很慢。可以在生成_book目录后，把其中的HTML文件和gitbook子目录（包含字体和js文件等）复制到自己的网站上。

Gitbook提供的push功能不能用。push.gitbook.io这个地址无法访问，不知是否是临时性服务故障。

Gitbook生成PDF的中文字体极其难看。万分期待改进。话说Gitbook生存的HTML上的中文非常漂亮。

在我的手机上看Gitbook的页面时，会让浏览器挂掉。

使用Gitbook发布电子书

上次说到用GitBook制作电子书，侧重在使用gitbook这个命令行工具，今天要说的重点是GitBook这个平台。当你把书放到GitBook上后，可以设置书的价格（每笔交易GitBook抽走20%作为佣金），也可以设置为免费，以及接受捐赠。如果你要收费或接受捐赠，则需要一个PayPal账户。在开始前，我要友情提示一句，在国内访问GitBook的速度很慢，通过VPN访问才好。

第**-1**步 用git这个源代码管理工具来管理你的Markdown文件。最好有个GitHub账户，这样每次push到GitHub时，GitBook都会自动为你的更新build新的版本（同时生成HTML、PDF、ePUB、MOBI这4个版本）。

第**0**步 注册一个GitBook帐号。

第**1**步 在GitBook添加一本书，填写书名等基本信息即可。完成后，GitBook会为你生成一个git仓库，其格式为：

> https://push.gitbook.io/{author}/{book}.git

`author` 即你的GitBook用户名， `book` 即你的书名，如我创建的书的git仓库：

> https://push.gitbook.io/berlinix/guaidanuniversity.git

这样你可以在编写完Markdown后，通过 `git push` 同步到GitBook。

第**2**步 把你本地的Markdown文件push到GitBook。我发现 `git push` 时常失败（服务器返回5xx错误），因此还有一种方法就是把你的GitHub项目与GitBook关联。每次push到GitHub时，会通过GitBook的webhook自动同步到GitBook上。

在Book Setting中简单配置一下即可，如我的配置为： `berlinix/gdu` （GitHub用户名为berlinix，GitHub仓库名为gdu）

在第一次push后，就可以看到你在GitBook上的电子书了，其访问地址为：

> http://{author}.gitbooks.io/{book}/

这是你电子书的主页，从这个页面可以直接打开HTML版本，或下载PDF等电子书版本，一般用户也可以为你的书添加评论。如：

> http://berlinix.gitbooks.io/guaidanuniversity/

要直接访问HTML版本，可以通过链接：

> https://www.gitbook.io/read/book/{author}/{book}

直接访问，如：

> https://www.gitbook.io/read/book/berlinix/guaidanuniversity

至此，GitBook平台的基本用法就介绍完毕。下面是我的一些使用经验。

个性化域名

HTML版本的URL很复杂，可以使用个性化域名简化之。在域名注册商那里添加一条CNAME记录即可，如：

> CNAME gdu.berlinix.com www.gitbook.io 300

并把 `gdu.berlinix.com` 配置到Book Setting中去，这样可以通过简单的 `gdu.berlinix.com` 来取代 `https://www.gitbook.io/read/book/berlinix/guaidanuniversity` 。同理，电子书的主页也可设置个性化域名，就不再赘述。

删除电子书

同样是在Book Setting中，可以删除电子书。在电子书列表中没有删除接口。

GitBook电子书封面

可以为电子书添加封面。只需添加2个名为 `cover.jpg` 和 `cover_small.jpg` 的两个图片即可。官方建议cover.jpg尺寸1800*2360，cover_small.jpg尺寸200*262。花2元即可在淘宝上找个做封面的人为你制造一个简单的封面，做得好就要花更多一些了 :)

GitBook帐号头像

似乎只接受Gravatar.com的头像。把Gravatar帐号关联过去即可。Gravatar提供的服务是把你的邮箱和头像关联起来，当你在其他网站注册时就不用每次都上传同一个头像，只需简单与Gravatar帐号关联即可。这样替换头像也方便了，一次替换、处处生效。

访问优化

按GitBook的访问速度，如果真让人访问GitBook上的HTML页面真是自寻死路啊，因此最好是把GitBook编译后的HTML放在自己的网站上。同时，为自己网站的HTML生成Disqus支持。例如你可以访问我放到自己服务器后的页面（用手机访问效果也非常好）：

> http://www.berlinix.com/gdu/index.html

电子书**Bug**

上次说到GitBook生成PDF的中文字体非常丑陋，另外还有一个问题，那就是生成的PDF可能是残全不全的。我编译后发现内容只剩一半。我已邮件过去报告这个Bug，还在等回信 :)

总体而言，GitBook还是很好玩，比起其他写作平台而言，要自由、简单，并舒服得多，可以用Vim编辑，支持Markdown语法，用git管理，关联GitHub后每次push后还能自动编译，生成多种电子书格式。如果你的书极为畅销的话，还能获取到捐赠或购买，没有理由不尝试的呀。

注

来源：

- 使用Gitbook制作电子书
- 使用GitBook平台发布电子书

Markdown + R

来源：Markdown写作浅谈

科技写作与Markdown+R

科技写作会碰到什么难题？

如果你是纯文科生，写的都是豆瓣小酸文或者诗歌之类的，那么，看完上面这一部分就可以打住了。如果你还有写科技论文的需要，则继续往下看。

科技写作与文艺写作的不同主要有：

- 公式与图表：相信各位写过科学论文的，都会为数学公式与各类图表的输出头疼不已；
- 格式转换：pdf是通用的，但是有时偏偏需要LaTeX原始格式或者Word原始格式；
- 参考文献：投稿给不同刊物，往往参考文献要根据对方的格式来调整。

解决这些难题，LaTeX是国际科学界，尤其是偏数理类的学科的主流方案之一。当然，因为中国盗版office的流行，导致国内科技论文Word更盛行，则是另一码事。Word因为近些年在参考文献协作软件、数学公式方面的发力，也逐步成为科技界认同的论文投递标准之一。

提到LaTeX的人们，常常有两种口气。一种是当做大神来敬仰的，当语言、软件变为传奇，路人皆知它的诞生历史时，于是，众多如你我这类文科生，只有抬头仰望的份了。另一类，则是不屑的口气，LaTeX那么好学，你怎么都学不会！国际期刊都是用这个写的，你别混了。。。

于是，我等文科生只好在被鄙视的眼光之下，快快走过LaTeX。。。但是，LaTeX真的符合人们写作习惯吗？请记住当时的历史。那时的计算机，所见即所得，并不像今天这么流行。那时的计算机，处理能力也不像今天这么强大。更别提什么脚本语言了。翻出上一份LaTeX文档所用的APA模版，大家就知道它有多么坑爹了。。。

apaexample.tex

排版 LaTeX 宏 标签 模板

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                           %
% Athanassios Protopapas, October 2005 %
% Mini-example for using apa.cls      %
%                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

\documentclass[man]{apa}

\title{Example of an APA-style manuscript}
\author{Athanassios Protopapas}
\affiliation{Institute for Language \& Speech Processing\\ Athens, Greece}

\abstract{This is an example of a minimal ``manuscript'' using the \LaTeX\ apa.cls document
class to typeset manuscripts according to the Americal Psychological Association (APA) manual
fifth edition.}

\acknowledgements{Written at the request of the Prac\TeX\ journal editors.

Comments may be sent to the author at protopap@ilsp.gr.}

\shorttitle{APA style manuscript}
\rightheader{APA style manuscript}
\leftheader{A.\ Protopapas}

\begin{document}
\maketitle
Here goes the text of the article. Note that the content begins immediately after
\texttt{maketitle} and there is no blank line between the title command and the article text. This
first section of the article is typically the introduction and, according to APA style, should not bear
a section heading.\footnote{That is, there is no ``Introduction'' section.} Subsequent sections,
however, are titled according to the psychological conventions.

\section{Experiment 1}
Manuscripts in APA style often contain descriptions of experiments. The APA manual
specifications for referring to experiments are to use a lowercase ``e'' when speaking generally,
as in the previous phrase, but an uppercase ``E'' when mentioning a particular experiment (as in
the following phrase), such as Experiment~1.

\subsection{Method}
The ``method'' is a subsection of the experimental presentation in which all the details of setting
up and conducting the experiment are described. There a number of more or less standard
components to a method, shown below.
\subsubsection{Participants}
Psychological experiments are conducted with participants, usually humans. Note that these
used to be called ``subjects'' but apparently APA now finds it inappropriate to refer to people
with this term. Here we mention how many participants there were, their ages and other
information about them.
\subsubsection{Apparatus}
Sometimes it is necessary to give the apparatus a special section.
\subsubsection{Stimuli}
What the subjects saw, heard, or felt.
\subsubsection{Procedure}
What happened to the poor subjects.
\subsection{Results}
In this subsection, one shows numbers and statistical analyses. Students are especially unlikely to
read this section, but seasoned researchers often avoid it in the first reading, especially if the
article is not of particular interest for their own research.
\subsection{Discussion}
If the results of the experiment mean anything, this is the place to talk about it.
```

使用Lyx，好看多了。问题是，它有坑吗？你跳过吗？

Markdown+R如何解决的？

每位试图解决LaTeX的不便，又试图保留它的优点的人们，都走上了一条不归路。

直到有一天，极其熟悉LaTeX，也熟悉Markdown的yihui同学，意识到了，LaTeX它可以作为最终格式生成。但是，我们中间的写作过程，完全可以用Markdown这么简单明了的语法来写，我们真正需要的，就是一堆数学公式、图表与参考文献而已。前2者，恰恰是R的强项。后者，则留给开源社区，下一步解决。（可参考线索1、线索2、线索 3 ）

于是，在他的新作R包knitr中，果断提供了Markdown支持。并说服R社区主流编辑器厂家，开源软件RStudio 提供 Markdown支持，从而使得Rmd这种新格式开始流行。我们有幸看到这个重要格式的诞生，国人的贡献如此重要。

Rmd 简介

Rmd 格式更详细的描述，读 yihui 的文档：自动化报告

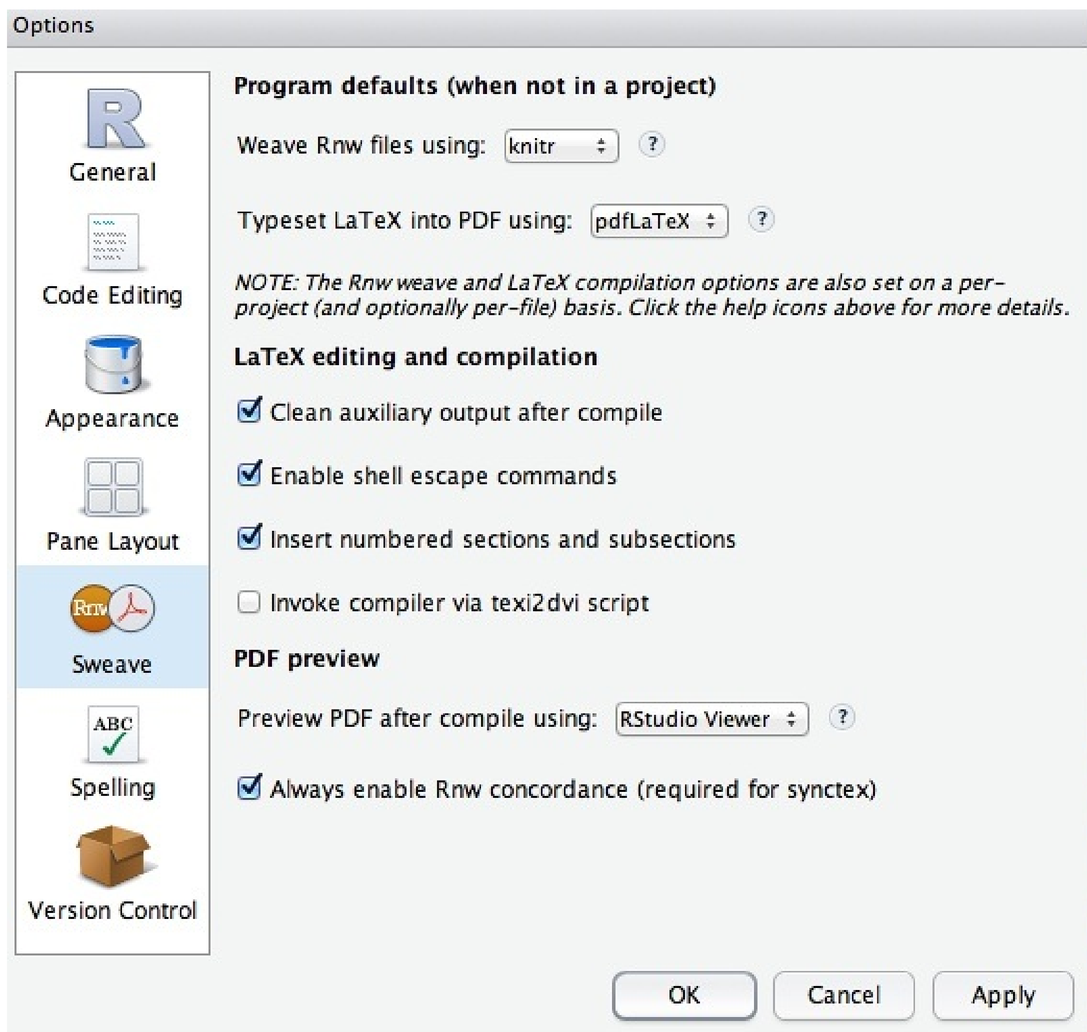
在这里，让我简单说明，如何最快上手Rmd格式。

安装并配置RStudio

下载 RStudio 之后，打开配置选项，如下图所示：

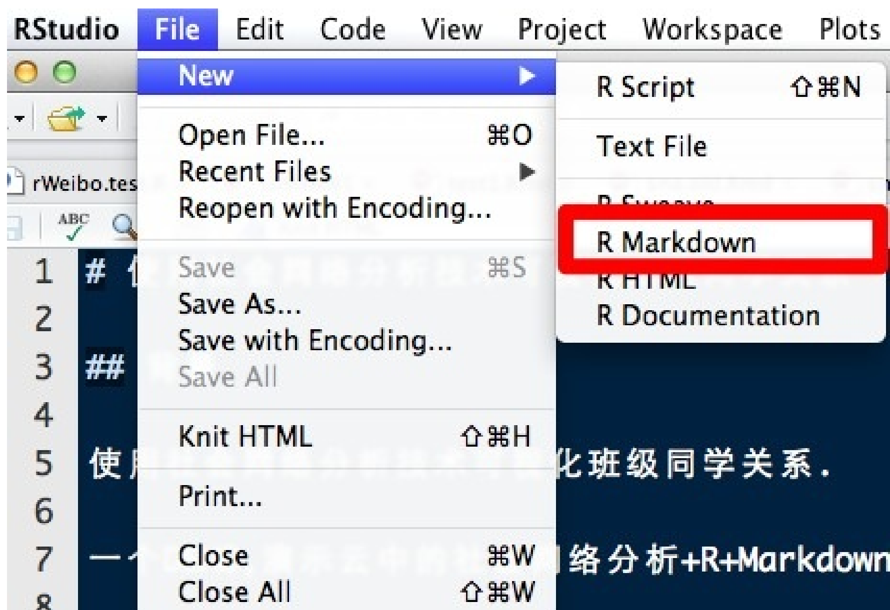


然后，进行如下配置：

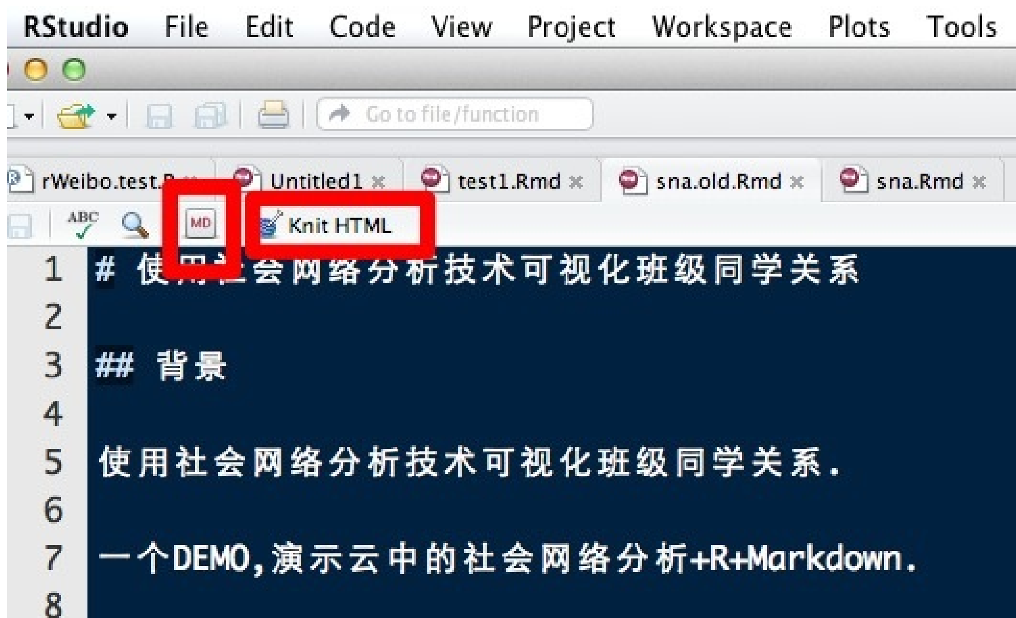


新建Rmd文档

新建一个Rmd文档，如下图所示：



然后，默认会出来一些内容。如果你对Markdown语法有不熟悉的地方，点击MD按钮。写完之后，直接点击： **Knit HTML** 按钮即可发布。MD按钮与Knit Html按钮的位置如下图所示：



就会预览成功。你也可以点击保存，生成相应的图片、Markdown文档。

是的，你要的一切图片都有了！这就是 yihui 所推崇的 文学性编程、可重复研究概念的神奇。

更重要的是，还保留了对LaTeX的无缝兼容。比如，大家可以敲下这段文字：

```
## The Normal Distribution

The normal distribution is defined as follows:

$$latex
f(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{ -\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2 }
$$

To generate random draws from a normal distribution we use the **rnorm** function:

```{r block1}
output <- rnorm(1000, 100, 15);
```

The normal distribution has the typical bell shape:

```{r block2, fig.width=8, fig.height=5}
ggplot2::qplot(output)
```
```

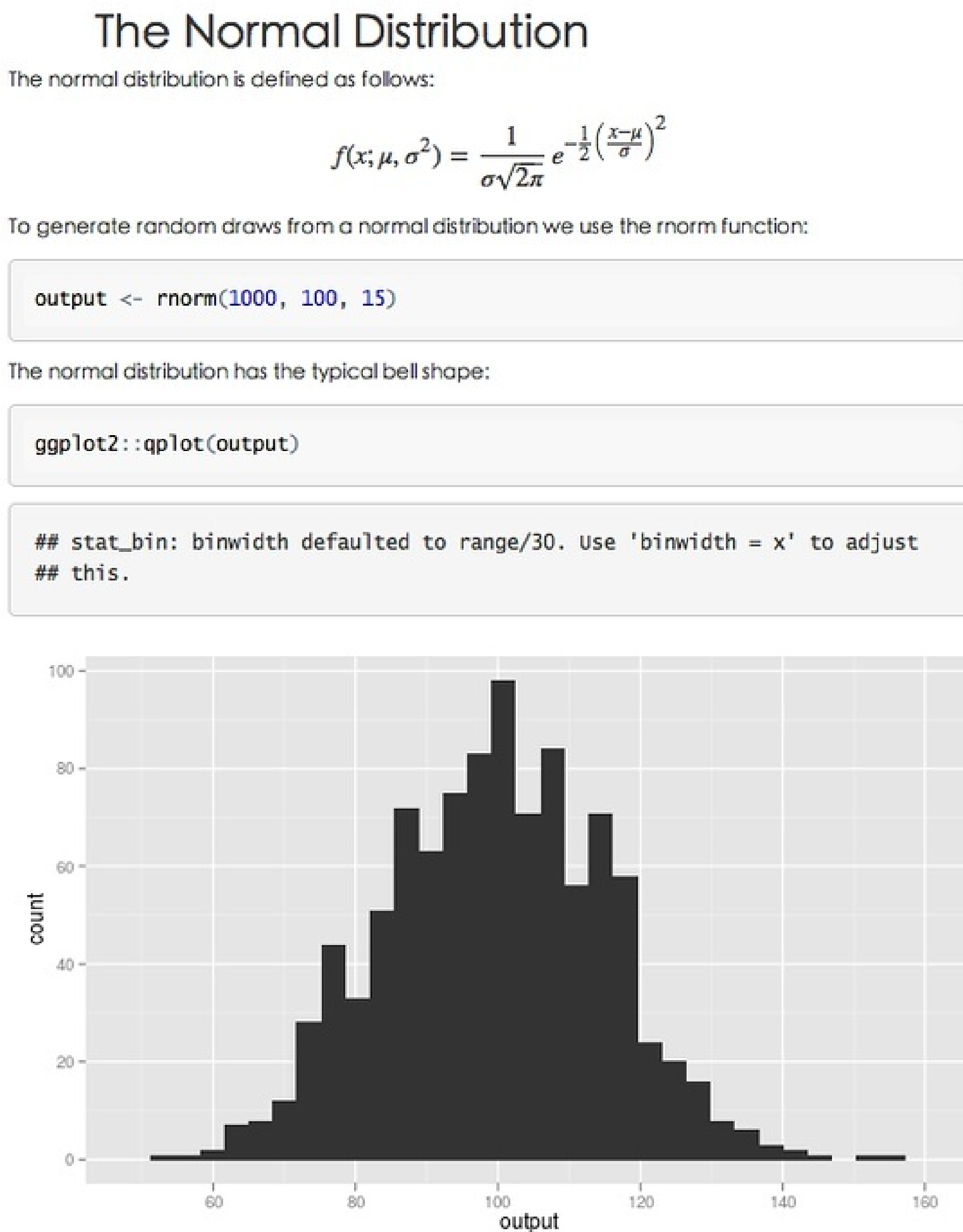
其中，这一段，

```
$$latex
f(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{ -\frac{1}{2}\left(\frac{x-\mu}{\sigma
$$
```

就是直接生成LaTeX格式的数学公式！

没有安装RStudio，或者不熟悉R的朋友，可以在我搭建的一个在线演示APP里面，将上述代码，粘贴上去，然后看看神奇的效果！

网址是：R Markdown App 效果如下图所示：



这么做，有什么好处呢？

让我细数一下：

真正意义上的可重复性研究

发表论文或者审核同事的报告，有个最麻烦的事情，你不知道他的步骤或者计算是否有误。现在，代码嵌在报告正文中，或者附录在报告末尾。而你，要做的，仅仅是一键生成。。。这就是真正意义上的可重复性研究！

更强大的数学与制图能力

既兼容了LaTeX的既有能力，同时，又广泛借助于R自身强大的作图与统计学习能力。

更重要的是，未来，并不是非要用R语言作图。yihui 同学在前文中的描述已经极其清楚了。

当然，还有云计算

真正意义上的云计算，尤其是类似于我们这样，中小企业、小型实验室实战使用的小型云计算，不同于各类忽悠的云计算。Markdown+R这种方式是最佳方式之一。上述例子中提到的那个APP，就是搭建在云中。同时提供各类REST接口，可以被Ruby程序调用。

Markdown格式与LaTeX、Word等格式的互转

点这里：Pandoc

还有不少有趣的玩法，如Building a beamer presentation with knitr.

如何学习Markdown+R？

好了，回到大家最关心的部分。分成两部分，先是如何学习Markdown，其次是如何学习R。

Markdown格式说明

- 参考：Markdown
- 更好的学习办法是直接读各类范本文件
- 更多资源参考V2ex节点

Markdown编辑器

- Mac等平台下推荐Mou

- Windows平台推荐MarkdownPad RStudio可作为写作科技论文与R语言编辑器选择，它是跨平台的
- markdown-here借助Chrome插件，将gmai写作窗口变为Markdown在线写作窗口
- 可以直接在线通过github撰写与提交Markdown文件，github有自动的版本跟踪功能，不用担心写废与找不到以前写的

Windows下的GitHub特别说明

- 如果碰到git、github等与windows不兼容的现象，不建议折腾，而是直接在线提交即可。
- GitHub最近发行了Windows版本，下载地址在这里
- 我的老文：如何高效利用github

如何学习R

Rstudio

- Getting_Started_with_RStudio.pdf

R语言入门读物

- R for SAS and SPSS Users.pdf ： 适合有SPSS基础的朋友
- Analysis of Questionnaire Data with R ： 适合处理问卷数据的文科生或社会科学类
- 更多参考我的豆列：技术派心理学

示范

文艺青年

文艺青年看这里，

- 為什麼文科生也該用markdown寫作?
- 为什么作家应该用 Markdown 保存自己的文稿

科学青年

可以看这里：如何学习科学：开放科学工具箱

点击 RAW 即可看到原始格式。这是一个长文档的示范。另一个示范是作者写的一个在线DEMO：

云中的社会网络分析+Markdown

以及：Markdown+R科技文写作

技术青年

- knitr以及各类Google、维基百科。
- 特别是：pandoc-markdown
- Primarily Pandoc: Writing in Markdown instead of LaTeX

Markdown + Pandoc

来源：Markdown+Pandoc，打通写作界的任督二脉！

Markdown+Pandoc，可以把自己的写作内容，变成世界上已有的任何格式的文件，包括很炫的slide，html5。没有人（或者我没看到）总结过这些内容，导致我走了很多弯路才最终打通任督二脉，特此纪念。

了解Markdwon以后，我的写作世界，只有它；看到Pandoc格式转换以后，对生成的slide和pdf羡慕的不行。那时，自己期望以后的写作是这样的：首先用Markdown把自己的想法写下来；其次，通过Pandoc，把写好的Markdown文件，转换成Slide或者PDF。如此而已。

Pandoc，这个不知道怎么发音，google也没找到。好吧，我就读做panda吧，谁让它是国宝。

Pandoc的运行，是在命令行里面。可是，没那么简单，不是任何一个cmd都可以。你必须要下载Pandoc，请参考这里。根据自己的os，选择Windows 或者其他。

安装以后，记得Pandoc的目录是啥，然后再到cmd里面去操作一些失传已久的doc命令，转换到pandoc的路径下。

我个人习惯，是把要转换的文件，比如test.md，放到pandoc的路径下，这样在使用pandoc转换的时候，不用输入太多的路径（尤其是我们很多路径是中文，怕可能有一些问题）。当然，也可以调用其他路径的文件，只要自己觉得舒服。

pandoc，就像linux下的iconv，可以把其他格式的文件，转化成自己想要的格式。具体的格式参考请看这里。

个人常用的有两个格式转换：

- a：md文件转换成html5

  ```
  pandoc -s --mathml -i -t dzslides test.md -o test.html
  ```

- b：md文件转换成pdf

  ```
  pandoc -t beamer test.md -o test.pdf
  ```

这里强调一点，如果想转成PDF文件，要安装LATEX。推荐安装MiKTex。但是，中文转PDF，因latex支持中文差，转换有问题。对于Latex熟悉的人，可以参考这个，看是否能解决中文转slide pdf的问题。

文件转换完成以后，如果有一些地方不合适，可以调整原始的md文件，再转换一次。等熟练以后，从写，到转换就非常迅速了。 当然，Pandoc还有很多的转换格式，大家可以自己去研究发觉。

关于Pandoc的使用，我没有过多的去研究。只是把自己常用的几个功能熟悉了一下。时间，真的真的很宝贵，不知不觉就从指缝中溜走了。所以，我只能在满足自己需求情况下，去使用pandoc。

期待大家更多的分享！

用markdown写博客：hexo + gitcafe

Hexo 入门指南（一） - 简介 & 准备

Hexo是一个开源的静态博客生成器，用node.js开发，作者是台湾大学生tommy351。

为什么是博客

对于个人网站来说，没有比博客更合适的形式了。在博客中，文章才是最主要的，一切都显得主次分明，干净利落。相比之下，论坛中主题和回复鱼龙混杂，阅读体验非常差。同时，博客比论坛的数据库小很多，便于维护。

为什么是静态博客

很多人选择在虚拟主机或vps上面搭建动态博客。但是这些主机商通常“免费的不稳定，稳定的不免费”。前一段时间，我观察了我的个人博客友链上面的几个站点，一部分在十几天之后就销声匿迹了。独立博客如此麻烦的维护工作，能不能减轻一些呢？正如阮一峰前辈所说，blogger分为三个阶段。最开始，是门户博客。之后，是独立博客。最后，觉得独立博客自己管理起来费劲，便找个别人来管的空间，自己负责写就好。如果我们能够找到这样的空间，在自己保留最大控制权前提下，由别人托管，会省去不少事情。

静态博客编译之后是纯html页面，优点就是支持它的环境十分好找，例如github、gitcafe、七牛云存储等站点都支持静态页面托管，自然是我们的首选了。由于github page在国内访问较慢，这篇文章用gitcafe做示范。gitcafe是天朝本地化的github，同样提供展示页和域名绑定功能，不需要备案，就是爽。

但是静态博客并非没有缺点。动态博客更新文章时，脚本是不变的，只需要更新数据库。静态博客要频繁改动文件，不支持增量式上传的东西，比如ftp，就难于管理。此外，还要十分熟悉git各种命令，才能部署页面。

准备工作

- git
- node.js
- markdown编辑器
- gitcafe
- 域名

markdown编辑器是非必须的，只要你熟悉语法，随便一个编辑器来写都不是问题。

域名也是非必须的，gitcafe pages服务提供免费的二级域名。注册域名的教程这里就不写了。

安装 git

git的客户端，本人推荐git-scm。

linux下面，在bash中键入：

(Ubuntu, Debian)

```
$ sudo apt-get install git
```

(Fedora, Red Hat, CentOS)

```
$ sudo yum install git
```

windows或mac下，直接到git-scm官网下载安装。

安装 node.js

linux下：

```
$ sudo apt-get install nodejs
$ sudo apt-get install npm
```

yum同理。

windows或者mac下，直接到node.js官网下载安装。 windows还要设置环境变量，把node.js安装路径写进path里面，用半角分号分隔。

markdown 编辑器

windows下推荐markdown pad。

mac下推荐mou。

gitcafe

首先注册一个账号，之后点击查看如何使用pages服务。

相关网页

- Hexo主页
- Hexo github 地址
- git book

Hexo 入门指南（二） - 安装、初始化和配置

安装和初始化

linux下打开bash，win下面打开cmd，输入：

```
$ npm install hexo -g
$ hexo init blog
$ cd blog
$ npm install
$ hexo server
```

访问http://localhost:4000，会看到生成好的博客。

同时，在blog文件夹中，文件如下：

```
2014/11/01  19:45    <DIR>          .
2014/11/01  19:45    <DIR>          ..
2014/11/01  11:16                68 .gitignore
2014/11/01  17:33            13,767 db.json
2014/11/01  11:16    <DIR>          node_modules
2014/11/01  11:17               186 package.json
2014/11/01  11:23    <DIR>          public
2014/11/01  11:16    <DIR>          scaffolds
2014/11/01  17:31    <DIR>          source
2014/11/01  11:16    <DIR>          themes
2014/11/01  11:38             1,844 _config.yml
```

配置

站点的配置文件是_config.yml，如果你不小心改花了，这里提供了一份默认的：

```
# Hexo Configuration
## Docs: http://hexo.io/docs/configuration.html
## Source: https://github.com/hexojs/hexo/

# Site
title: Hexo
subtitle:
description:
author: John Doe
email:
language:

# URL
## If your site is put in a subdirectory, set url as 'http://yoursite.com/child' and root
url: http://yoursite.com
root: /
permalink: :year/:month/:day/:title/
tag_dir: tags
archive_dir: archives
category_dir: categories
code_dir: downloads/code
permalink_defaults:
```

```
# Directory
source_dir: source
public_dir: public

# Writing
new_post_name: :title.md # File name of new posts
default_layout: post
titlecase: false # Transform title into titlecase
external_link: true # Open external links in new tab
filename_case: 0
render_drafts: false
post_asset_folder: false
relative_link: false
highlight:
  enable: true
  line_number: true
  tab_replace:

# Category & Tag
default_category: uncategorized
category_map:
tag_map:

# Archives
## 2: Enable pagination
## 1: Disable pagination
## 0: Fully Disable
archive: 2
category: 2
tag: 2

# Server
## Hexo uses Connect as a server
## You can customize the logger format as defined in
## http://www.senchalabs.org/connect/logger.html
port: 4000
server_ip: localhost
logger: false
logger_format: dev

# Date / Time format
## Hexo uses Moment.js to parse and display date
## You can customize the date format as defined in
## http://momentjs.com/docs/#/displaying/format/
date_format: MMM D YYYY
time_format: H:mm:ss

# Pagination
## Set per_page to 0 to disable pagination
per_page: 10
pagination_dir: page

# Disqus
disqus_shortname:

# Extensions
## Plugins: https://github.com/hexojs/hexo/wiki/Plugins
## Themes: https://github.com/hexojs/hexo/wiki/Themes
theme: landscape
exclude_generator:

# Deployment
## Docs: http://hexo.io/docs/deployment.html
deploy:
  type:
```

官方的页面上也提供了每一项详细的解释。

我们需要修改的只有Site部分，以及URL部分的url。Site部分每一项依次是标题、副标题、描述、作者、邮箱和语言（天朝大陆填zh-CN）。url改成网站的网址，如果你的网站放在某个子目录下，比如http://yoursite.com/child，root改成/child。

Server部分，如果之前你的服务器没有运行起来，则可能是端口被占了。把port改成别的数字，或者强行关掉占着端口的进程。

其它设置项先不用管，将会在接下来的文章中解释。

注意

如果页面中出现中文，应以**UTF-8**无**BOM**编码格式，所以不要用**win**自带的记事本，而是用**notepad++**这种支持编码转换的编辑器。

由于google在天朝大陆被墙，进入themes\landscape\layout_partial，打开head.ejs，删掉第31行fonts.googleapis.com的链接。

下载下来jquery-2.0.3.min.js，放到themes\landscape\source\js文件夹中。之后进入themes\landscape\layout_partial，打开after-footer.ejs，将第17行的路径替换为/js/jquery-2.0.3.min.js。

至此大功告成。

Hexo 入门指南（三） - 文章 & 草稿

文章

命令行中输入：

```
$ hexo new "new article"
```

之后在source/_posts目录下面，多了一个new-article.md的文件。

打开之后我们会看到：

```
title: new article
date: 2014-11-01 20:10:33
tags:
---
```

文件的开头是属性，采用统一的yaml格式，用三条短横线分隔。下面是文章正文。

文章的正文支持markdown格式，建议你先学习一下它的语法。markdown不像html似的一大堆标签，很简单，只有几个符号。

新建、删除或修改文章后，不需要重启hexo server，刷新一下即可预览。

属性

文章可以拥有如下属性：

Setting	Description	Default
layout	Layout	post或page
title	文章的标题	
date	创建日期	文件的创建日期
updated	修改日期	文件的修改日期
comments	是否开启评论	true
tags	标签	
categories	分类	
permalink	url中的名字	文件名

动态博客中通过发布文章页面设置的各种属性，在hexo里要这样设置。

分类和标签

例如：

```
categories:
- 日记
tags:
- Hexo
- node.js
```

摘要

同wordpress一样，之上的内容为摘要。

layout

如果你修改了layout，在scaffolds文件夹里一定要有名字对应的模版文件，否则会采用默认模版。

文件名

在配置文件中的new_post_name项可以设置文件名，默认为:title，也就是你在命令行输入的名字。

文件名可以为下面几个变量和字符串常量的任意组合：

Variable	Description
:title	Escaped title (lower case and replace spaces with dash)
:year	Created year (4-digit)
:month	Created month (2-digit)
:i_month	Created month (Without leading zeros)
:day	Created day (2-digit)
:i_day	Created day (Without leading zeros)

草稿

草稿相当于很多博客都有的“私密文章”功能。

```
$ hexo new draft "new draft"
```

会在source/_drafts目录下生成一个new-draft.md文件。但是这个文件不被显示在页面上，链接也访问不到。也就是说如果你想把某一篇文章移除显示，又不舍得删除，可以把它移动到_drafts目录之中。

如果你希望强行预览草稿，更改配置文件：

```
render_drafts: true
```

或者，如下方式启动server：

```
$ hexo server --drafts
```

下面这条命令可以把草稿变成文章，或者页面：

```
$ hexo publish [layout] <filename>
```

Hexo 入门指南（四） - 页面、导航、边栏、底栏

页面

命令行键入：

```
$ hexo new page about
```

会在source/about中生成index.html。这个就叫做页面，不在文章列表显示，可以通过http://localhost/about浏览。

页面支持文章的大部分属性，除了分类和标签。

导航

打开主题中的设置文件，即themes_config.yml（其中是当前主题的名字，默认为landscape，下同），找到menu:，在列表的末端添加About: 关于。刷新页面，导航栏上就出现了关于链接。

边栏

进入themes\\layout_widget目录中，创建about.ejs文件，模仿其他文件中的模版，输入以下内容：

```
<% if (site.tags.length){ %>
  <div class="widget-wrap">
    <h3 class="widget-title">About</h3>
    <div class="widget">
      邮箱：xxx@xxx.com<br />
      微博：@xxxxx
    </div>
  </div>
<% } %>
```

打开themes_config.yml，找到#Sidebar，在最后面添加- about。刷新页面。

底栏

打开themes\\layout_partial\footer.ejs修改。

-banner-

打开themes\\source\css\images，把banner.jpg换掉。

Hexo 入门指南（五） - 搬家 & 备份

搬入hexo

首先，需要拿到原博客数据的xml文件。

wordpress的话，后台“工具->导出”就可以生成。点点和lofter也支持类似操作。如果遇到不支持导出xml的博客，先用http://www.diandian.com/transfer/转到点点，再用http://www.diandian.com/backup导出XML文件。

之后，安装hexo-migrator-wordpress这个插件

```
npm install hexo-migrator-wordpress --save
```

运行

```
hexo migrate wordpress wordpress.xml
```

xml中的数据就导入到source中了。最后的工作是修复链接什么的。

搬出hexo

没有什么好的办法。可以写个脚本遍历public文件夹，之后post到指定目录或者制作成xml文件。

备份

有句话说得好，数据恢复的最佳方案永远是“备份备份再备份”。

个人建议，分别备份站点配置和文章。站点配置包括blog根目录除了source和public文件夹的所有内容，文章就是source文件夹的全部内容。站点配置不经常变的话可以不用经常备份。

Hexo 入门指南（六） - sitemap、rss 和部署

sitemap & rss

切换到blog根目录下，输入：

```
$ npm install hexo-generator-feed
$ npm install hexo-generator-sitemap
```

之后重启博客，访问/atom.xml和/sitemap.xml，会发现已经生成了。可以把sitemap提交到搜索引擎的站长平台来增加收录。

部署

首先按照前面教程（一）的gitcafe部分建立好代码仓库，这里假设你的用户名是your_name。由于ssh配置比较麻烦，这里采用https方式提交。

找到配置文件中# Deployment一节，修改：

```
type: github
repository: https://gitcafe.com/your_name/your_name.git
branch: gitcafe-pages
```

之后输入：

```
hexo deploy --generate
```

或者

```
$ hexo generate --deploy
```

hexo会自动生成并部署。

如果之前已经生成过了，直接输入：

```
$ hexo deploy
```

部署即可。

当然，这个命令还有一些bug，比如windows下不能用cmd而是用gitshell。我自己一般会手动敲git代码覆盖提交。

Hexo 入门指南（七） - 评论 & 分享

评论

hexo默认集成了disqus，但是在天朝明显多说更受欢迎一点。

首先到多说官网去注册一个账号。然后点击进入添加站点页面，填写所有信息。注意，多说域名的前缀就是站点的短网址，下面要用到，这里假设为short_name。

在_config.yml中添加多说的配置：

```
duoshuo_shortname: short_name
```

修改themes\\layout_partial\article.ejs，把第38行到41行的如下代码：

```
<% if (!index && post.comments && config.disqus_shortname){ %>
<section id="comments">
  <div id="disqus_thread">
    <noscript>Please enable JavaScript to view the <a href="//disqus.com/?ref_noscript">c
  </div>
</section>
<% } %>
```

替换成：

```
<% if (!index && post.comments && config.duoshuo_shortname){ %>
<section id="comments">
  <!-- 多说评论框 start -->
  <div class="ds-thread" data-thread-key="<%= post.layout %>-<%= post.slug %>" data-title
  <!-- 多说评论框 end -->
  <!-- 多说公共JS代码 start (一个网页只需插入一次) -->
  <script type="text/javascript">
  var duoshuoQuery = {short_name:'<%= config.duoshuo_shortname %>'};
    (function() {
      var ds = document.createElement('script');
      ds.type = 'text/javascript';ds.async = true;
      ds.src = (document.location.protocol == 'https:' ? 'https:' : 'http:') + '//static.
      ds.charset = 'UTF-8';
      (document.getElementsByTagName('head')[0]
       || document.getElementsByTagName('body')[0]).appendChild(ds);
    })();
  </script>
  <!-- 多说公共JS代码 end -->
</section>
<% } %>
```

之后，找到第27到29行：

```
<% if (post.comments && config.disqus_shortname){ %>
  <a href="<%- post.permalink %>#disqus_thread" class="article-comment-link">Comments</a>
<% } %>
```

替换成：

```
<% if (post.comments && config.duoshuo_shortname){ %>
  <a href="<%- url_for(post.path) %>#comments" class="article-comment-link">留言</a>
<% } %>
```

分享

hexo默认提供的那四个在国内也被墙了。这里替换成百度一键分享。

找到themes\landscape\layout_partialarticle.ejs26行：

```
<a data-url="<%- post.permalink %>" data-id="<%= post._id %>" class="article-share-link">
```

替换成：

```
<a data-url="<%- post.permalink %>" data-id="<%= post._id %>" class="article-share-link b
<script>window._bd_share_config={"common":{"bdSnsKey":{},"bdText":"","bdMini":"1","bdMini
```

之后打开themes\landscape\source\js\script.js，35~86行全部注释掉。

后记

仅以此教程，悼念Aaron Swartz，RSS和Markdown的联合创始人。没有他，开源博客界就不会有今天。